Universidade do Minho
Escola de Engenharia | Departamento de Informática
Licenciatura em Engenharia Informática

Unidade Curricular de Bases de Dados

Ano Letivo de 2022/2023

Bomba de Gasolina

Grupo 32
Jorge Emanuel Matos Teixeira (a100838) José Luís Fraga Costa (a100749)
José Pedro Batista Fonte (a91775) Miguel Velho Raposo (a94942)

26 de setembro de 2023

BD

Data de Receção	
Responsável	
Avalição	
Observações	

Bomba de Gasolina

Grupo 32
Jorge Emanuel Matos Teixeira (a100838) José Luís Fraga Costa (a100749) José Pedro Batista Fonte (a91775) Miguel Velho Raposo (a94942)

26 de setembro de 2023

Resumo

O presente relatório tem como caso de estudo uma bomba de gasolina onde se pretende implementar um sistema de base de dados que auxilie numa melhor gestão de stocks, vendas e funcionários.

Área de Aplicação: Gestão de Bases de Dados, Gestão de loja, Bomba de Combustível

Palavras-Chave: Bases de Dados, SQL, Modelo Lógico, Microsoft PowerBi,

Índice

Lista de Figuras

Lista de Tabelas

1 Definição do Sistema

1.1 Contextualização

O caso em estudo trata do Sr.Domingos Celso que vive em Miranda do Douro, numa aldeia remota onde predomina a exploração agrícola. Dada a sua localização geográfica sempre que o Sr.Domingos Celso e os restantes habitantes precisam de reabastecer as suas máquinas (agrícolas e pessoais), é necessário percorrer longas distâncias até cidade mais próxima, ou até Espanha, sujeitando-se à inconveniência de várias horas de deslocação e gastos acrescidos.

Devido a sua vertente empreendedora o Sr.Domingos decidiu construir uma bomba de gasolina que abasteça os habitantes locais e os viajantes que passam pela aldeia. Na sua descrição mais simples, o seu novo empreendimento integra uma parte de venda de combutiveis e uma parte de loja de conveniência, onde comercializa produtos como bebidas, snacks, produtos para carros, entre outros.

Após os primeiros meses de trabalho o negócio cresceu rapidamente, levando o Sr.Domingos a contratar alguns funcionários, no entanto, apercebeu-se que não era suficiente pois o seu modo analógico de gestão da bomba, com livro de registos e "papelada"tradicional, era incapaz de satisfazer o grande fluxo de informação diário, e isso estava claramente a compremeter a qualidade do serviço. Desse modo decidiu implementar um sistema digital que gere a parte de loja, de combustiveis e de funcionários e para implementar esta solução, o Sr.Domingos falou com a nossa equipa de trabalho e descreveu todos os processos envolventes no funcionamento da bomba.

Numa descrição mais pormenorizada o Sr.Domingos explicou que a bomba de gasolina apresenta duas frentes de negócio: a venda de combustiveis e a venda de produtos numa loja de conveniência.
Na parte da venda de combustiveis, a bomba comercializa gasóleo simples, gasóleo aditivado, gasóleo agrícola, gasolina 95 e gasolina 95 aditivada em 4 postos de abastecimento. Quanto à compra dos combustiveis, todos os clientes tem à sua disposição todas as opções mencionadas anteriormente, à exceção do gasóleo agrícola, que é necessário a apresentação de um cartão agrícola no ato da compra. Em termos legais, o Sr.Domingos mantém um registo das quantidades de combustível de cada tanque no início e fim do dia, e os valores tem de estar de acordo com as vendas ao longo do dia. Outro pormenor que também mencionou, é que é obrigatório por lei, 10% da capacidade do tanque do combustivel estar vazio. Quando o tanque começa a esvaziar o Sr.Domingos envia um pedido de reabastecimento ao fornecedor,

especificando o produto e a quantidade, no entanto, o fornecedor impõe um limite mínimo de 25 mil litros de combustivel total (podendo ser um ou mais tipos diferentes).

Na parte da loja o Sr.Domingos vende bebidas, snacks, produtos de carros, entre outros. Para manter um controlo operacional sobre a loja existem vários livros de registos: o Livro de Registo de Vendas, o Livro de Registo de Stocks e o Livro de Registo de Encomendas.

As entradas no livro de vendas acontecem no momento de cada venda, onde se guarda uma cópia da fatura manual emitida pelo funcionário. Uma fatura inclui informação sobre os produtos, o seu preço, o iva, a quantidade, os descontos, o valor total, o método de pagamento, as informações da empresa, as informações da fatura. Durante o dia os funcionarios vão atualizando o livro de stocks com as quantidades restantes de cada produto.

No livro de encomendas regista-se a toda a informação sobres os forncedores e sobre as encomendas como : a lista de produtos, o seu preço de retalho, as quantidades, o fornecedor, a data da encomenda e a data de entrega expectável, ficando a encomenda com o estado "Pendente". No momento de entrega verifica-se se toda a encomenda está ou não correta e regista-se a data de entrega, sendo que o estado da encomenda é alterado para "Entregue". Nos casos de cancelamento da encomenda, regista-se o estado como "Cancelada".

Na parte de gestão dos funcionários o Sr.Domingos mantém um arquivo, que só ele tem acesso, com os contratos dos funcionários e todas as informações pessoais. Cada funcionário ofere diferentes salários de acordo com função que cumpre na bomba, o seu horário semanal e os seus anos de trabalho. Na bomba existem duas funções, a de gerente e a de operário. Os funcionários diariamente tem de cumprir quatro tipos de tarefas: recolher encomendas, registar as vendas, controlar stock e abastecer viaturas. O gerente cumpre as mesmas tarefas dos funcionários e tem acrescido as funções de contactar os fornecedores, pagar aos funcionários e arranjar novos clientes.

Com o novo sistema o Sr.Domingos pretende libertar carga laboral sobre os seus funcionários em funções como controlo de stocks e registo de vendas, de modo a ser capaz de oferecer um novo serviço de entrega de combustivel ao domicílio. Este novo serviço necessita de um funcionário especializado no transporte de matérias perigosas e uma viatura especializada. Para isso o Sr.Domingos pretende implementar um sistema de registo dos seus clientes e as suas informações relevantes.

1.2 Fundamentação

As principais dificuldades que o Sr.Domingos detetou no momento de expansão do seu negócio e que o levaram a implementar um novo sistema foram:

- **Gestão da Informação** : Com o acumular do tempo os seus livros de registo tornaram-se um mau método de armazenar, manipular a acessar informação. O cruzamento de informação entre livros de registos tornou-se muito demorada, pouco organizada e

propensa a erros.

- **Falta de Estatísticas** : A dificuldade de gerir informação impede o Sr.Domingos de fazer previsões futuras baseadas em dados empíricos, levando a periodos de falta ou excesso de stock. A falta de estatísticas facilmente acessiveis também prejudica a avaliação da gestão da bomba e da performance dos funcionários.

- **Cumprimento de medidas legais** : As bombas de gasolina estão sujeitas a medidas legais apertadas quanto à venda de combustiveis, emissão de faturas, registo de niveis de combustiveis. os possíveis erros no preenchimento da burocracia associada a estas medidas leva a multas pesadas para a bomba.

- **Baixa Produtividade** : Segundo um estudo levado a cabo pela equipa, a bomba emprega mais 35% dos funcionários do que o que realmente necessita. A baixa produtividade de cada funcionário deve-se ao método de registo nos Livros de Registos e ao acesso complicado e demorado a informação relevante.

- **Medição do nivel de combustivel** : Cada vez que seja necessário medir o combustivel, um funcionário tem de abrir o tanque, colocar uma bareta de medição e registar o nivel medido.

- **Filas de Espera** : Em dias de muita afluência à bomba o método de emissão manual de faturas resulta em filas de espera para os pagamentos e, por consequente, pior qualidade de serviço.

Identificados todos estes problemas o Sr.Domingos vê num Sistema de Base de Dados uma solução capaz de os resolver, automatizando e agilizando processos cada uma das áreas mencionadas.

1.3 Motivação e Objetivos

A motivação principal é desenvolver um sistema de base de dados que auxilie o Sr.Domingos numa melhor gestão do seu novo empreendimento, fornecendo uma forma mais rápida e eficiente de gestão de stock, vendas e de funcionários.
Com essa motivação em mente a equipa de trabalho traçou os seguintes objetivos para o projeto:

- Registo de compra/venda de produtos e combustíveis
- Organização e Monitorização dos stocks dos produtos
- Previsão antecipada da falta de stock
- Entender melhor os clientes com a análise de tendências de vendas

- Aumentar a produtividade
- Automatização de procedimentos legais
- Reduzir o tempo gasto em burocracia
- Melhorar a qualidade de serviço da bomba
- Organização de contratos dos trabalhadores
- Integração com os fornecedores
- Implementar um serviço de entrega ao domicilio
- Implementar um registo de clientes

1.4 Viabilidade

Em conjunto com a equipa de trabalho o Sr.Domingos espera que o seu novo sistema ao substituir o antigo livro de registos e papéis auxiliares, permita o seguintes ganhos:

- Melhorar a gestão de stocks dos produtos em 35%, reduzindo os gastos mal calculados.
- Melhorar a gestão de funcionários, libertando carga laboral e reduzindo gastos em funcionários em 35%. Por acréscimo, reduzir os gastos em papel em 70%.
- Oferecer uma nova vertente de negócio (entrega ao domícilio), aumentando as receitas em 35%.
- Melhorar a qualidade do serviço em 135%.
- Permitir a capacidade de uma expansão futura sem compromenter a qualidade de serviço.

1.5 Recursos e Equipa de Trabalho

Os recursos necessários à implementação e desenvolvimento de um Sistema de Base de Dados são:

- **Humanos**
  - Sr.Domingos, Funcionários da Loja, Fornecedor
  - Equipa de trabalho da empresa de desenvolvimento

- **Materiais**
  - Hardware : 1 Servidor, 1 posto de venda de loja, 4 postos de abastecimento, Sensores de Capacidade de Tanques;
  - Software : Software de Vendas, Sistema de Gestão de Bases de Dados.

A equipa de trabalho será composto por:

Pessoal Interno

- **Sr.Domingos Celso, Funcionários e Fornecedor :** Para entender o funcionamento da bomba, atendimento a clientes e o fornecimento de encomendas.

Pessoal Externo

- **Gestor de Projeto :** Responsável por supervisionar todo o projeto e garantir que ele é concluído dentro do prazo, do orçamento e de acordo com a satisfação do cliente. Também é responsável por coordenar a equipa de trabalho de acordo com o trabalho desenvolvido e o feedback do cliente.
- **Arquiteto de Bases de Dados :** Responsável por projetar o esquema da base de dados e determinar como os dados serão armazenados, organizados, manipulados e acessados. Este irá trabalhar em estreita colaboração com o cliente para entender suas necessidades e requisitos e garantir que a base de dados seja projetado para atender as suas expectativas.
- **Engenheiro de Bases de Dados:** Responsável por implementar o design e construir o sistema da base de dados, assim como, manter a manutenção do sistema ao longo do seu tempo de vida.
- **Engenheiro de Testes e Garantia da Qualidade :** Responsável por garantir que o sistema construido está de acordo com as especificações do cliente e que mantém a qualidade de performance pretendida. Para tal, terá de testar o sistema minuciosamente tentando encontrar bugs, erros ou problemas de performance e trabalhar em conjunto com a equipa de desenvolvimento para os resolver.

A equipa de trabalho implementa as funcionalidades pretendidas com as seguintes ferramentas de software:

- `Gestão e Apresentação do Projeto` : Github, Overleaf, Google Docs e PowerPoint
- `Levantamento e Análise de Requisitos` : Google Docs e Excel
- `Modelação Conceptual dos Dados` : BrModelo
- `Modelação Lógica` : MySQL WorkBench

- `Servidor de Bases de Dados` : MySQL Community Server
- `Visualização e Análise de Dados` : Power BI

1.6 Plano de Execução do Projeto

De modo a organizar e gerir o desenvolvimento do projeto, a equipa de trabalho elaborou um plano específico e concreto de execução do projeto baseado num Diagrama de Gant. O plano inclui todas as fases de um ciclo de vida de um SBD definindo as respetivas tarefas e os seus responsáveis.

[Diagrama de Gant]

1.7 Análise e Validação do Sistema

A fase de "Definição do Sistema"do projeto foi desenvolvida em proximidade com o Sr.Domingos. Esse modo de trabalho permitiu detalhar as operações do seu negócio, definir objetivos e expectativas para o projeto e traçar um plano de desenvolvimeto do mesmo, onde se apresentou a equipa de trabalho e se estabeleceu prazos de entrega.

Alguns exemplos das várias reuniões com o Sr.Domingos foram:

- Contextualização : onde se detalhou o *modus operandi* da sua empresa.
- Motivação e Objetivos : onde se definou alvos claros atingir no final do desenvolvimento.
- Viabilidade : onde se estabeleceu as métricas de sucesso esperadas após a implementação de um SGBD.

Após uma reunião final de validação do sistema a resposta foi positiva e a equipa avançou para a fase seguinte - o Levantamento e Análise de Requisitos.

2 Levantamento e Análise de Requisitos

2.1 Método de Levantamento e Análise de Requisitos

A metodologia adotada passa por uma execução sequencial de procedimentos: 1º levantamento, 2º organização e 3º revisão e validação. O levantamento de requisitos focou-se em 4 tipos diferentes:

- Reuniões e entrevistas : permitem recolha rápida e personalizada da informação, a validação do trabalho desenvolvido e a identificação de novos requisitos. Alguns exemplos dos resultados deste método foram:
  - Definir as permissões de emissão de novas encomendas.
  - Conhecer os diferentes combustivéis e produtos comercializados.
  - Perceber as informações relevantes sobre cada cliente.
  - Entender o tipo de consultas ao sistema que se espera.
- Análise dos Livros de Registos utilizados : permite o levantamento de parâmetros importantes na gestão da bomba. Alguns exemplos dos resultados deste método foram:
  - Livro de Registos de Stock : saber a informação registadas por cada produto.
  - Livro de Registos de Vendas : saber a informação de cada fatura emitida.
  - Livro de Registos de Encomendas : saber a informação de cada encomenda e a informação dos fornecedores.
- Observação dos processos de trabalho da bomba no dia a dia : permite a associação de entre entidades do sistema, identificação de novos requisitos e contexto sobre o sistema a ser implementado. Alguns exemplos dos resultados deste método foram:
  - Perceber dados relevantes à operação da bomba, que depois se pretende tornar disponíveis - "Consulta de vendas por funcionário", "Consulta do lucro de vendas por intervalo de tempo", "Consulta de quantidades".

De notar, que a estrutura de cada requisito é : o requisito, data e hora, quem levantou(fonte), quem forneceu(analista), ârea de aplicação e revisor.

Deste modo, foi possível encontrar uma base sólida de evidências e observações que fundamentam o levantamento dos requisitos.

2.2 Organização dos Requisitos Levantados

Os requisito estão organizados de acordo com o seu efeito na Base de Dados, podendo ser de três tipos:

- Requisito de Descrição : acolhe os requisitos que referem a criação de objetos na base de dados.
- Requisito de Exploração : acolhe os requisitos que definem como se insere/atualiza/remove/consulta dados do sistema.
- Requisito de Controlo : acolhe os requisitos que definem as permissões de inserção, manipulação e consulta pelos diferentes utilizadores.

Dentro de cada um foram identificadas 4 áreas diferentes da gestão da bomba, sendo elas:

- Gestão de Stocks : Registo, Monitorização e Controlo do stock de produtos e combustiveis, assim como os seus fornecedores e as respetivas encomendas.
- Gestão de Vendas : Registo e estatísticas das vendas.
- Gestão de Funcionários : Contratos e informações relavantes dos funcionários.
- Gestão de Clientes : Registo e estatísticas dos clientes.

Em anexo, encontra-se a estrutura completa do levantamento de requisitos, que inclui informação adicional como a área, fonte, o analista e o revisor de cada requisito.

2.2.1 Requisitos de Descrição

Gestão de Stocks

- O sistema mantém um registo de todos os produtos/combustíveis comercializados na bomba, assim como os seus fornecedores e as respetivas encomendas.
- Cada produto da loja deve incluir um identificador único, nome, marca, quantidade, preço PVP, categoria (bebidas, snacks, artigos carro,etc) e o seu fornecedor .

- Cada combustível está associado a um identificador único, nome, quantidade no tanque (em litros), preço, IVA e fornecedor.
- Cada fornecedor tem identificador único, nome, endereço, informação fiscal, contacto e histórico de encomendas feitas.
- Cada encomenda deve ter uma lista de produtos (com preço de retalho e quantidade), valor total da encomenda, data da encomenda, data prevista de entrega, data de entrega e estado da encomenda (Entregue, Pendente e Cancelada).

Gestão de Vendas

- O sistema mantém um registo das vendas através das faturas emitidas.
- Cada fatura possui um identificador único, hora e/ou data, método de pagamento, dados da bomba, dados do funcionário, dados do cliente e a lista de produtos, desconto total e preço total.
- Cada produto na fatura contém o nome do produto, preço, iva, quantidade, valor total, desconto do produto.
- Os dados do funcionário na fatura são o seu id e nome.
- A bomba tem a si associada um nome, uma morada, contacto e um NIF.
- Na emissão de uma fatura a associação a um cliente é opcional.

Gestão de Funcionários

- O sistema mantém um registo dos funcionários que emprega.
- Cada funcionário tem um identificador único, nome, data de nascimento, função que exerce na bomba, data de inicio e duração do contrato, salário, horário semanal, informação fiscal, informações de pagamento, informações de contacto (telemóvel, e-mail).

Gestão de Clientes

- O sistema mantém um registo dos seus clientes, que optam por dar as suas informações. Para serviços de entrega ao domicilio é um requisito obrigatório.
- Cada cliente tem identificador único, nome, NIF, data de registo, endereço(opcional), e-mail(opcional), nº de telefone(opcional) e as suas compras realizadas na bomba.

2.2.2 Requisitos de Exploração

Gestão de Stock

- **Inserção**
  - Novos produtos podem ser inseridos especificando todas as suas informações.
  - Novos combustíveis podem ser inseridos especificando todas as suas informações.
  - Novos fornecedores podem ser inseridos especificando todas as suas informações.
  - Novas encomendas podem ser criadas especificando todas as suas informações.
  - Ao encomendar combustível existe um limite mínimo de 25000 litros por encomenda.
- **Atualização/Remoção**
  - Atualização/Remoção de um produto.
  - Atualização/Remoção de um combustível.
  - Atualização/Remoção de um fornecedor.
  - Atualização do estado de uma encomenda.
  - Sempre que uma encomenda é dada como "Entregue"a lista de produtos/combustíveis deve ser atualizada acrescentando as novas quantidades.
  - Quando é realizada uma venda o stock dos produtos é alterado.
- **Consulta**
  - Consulta das informações de um produto.
  - Consulta da previsão de falta de stock de um certo artigo.
  - Consulta ordenada do stock de todos os produtos.
  - Consulta das informações de um combustivel.
  - Consulta ordenada da quantidade de todos os combustíveis.
  - Consulta das informações de um fornecedor.
  - Consulta das informações de uma encomenda.
  - Consulta de todas as encomendas por fornecedor.
  - Consulta de todas as encomendas num dado intervalo de tempo.
  - Consulta ordenada dos maiores fornecedores.

- Consulta das encomendas "Pendentes".

Gestão de Vendas

- **Inserção**
  - Emissão de novas faturas a cada venda.
- **Atualização/Remoção**
  - Atualização/Remoção de uma venda.
- **Consulta**
  - Consulta das informações de uma fatura.
  - Consulta do valor total de vendas por intervalo de tempo, p.e.: mês, trimestre, ano.
  - Consulta do lucro de vendas por intervalo de tempo, p.e.: mês, trimestre, ano.
  - Consulta de vendas por funcionário.
  - Consulta das vendas de um produto/combustivel.

Gestão de Funcionários

- **Inserção**
  - Contratação de novos funcionários e associação de todas as suas informações.
- **Atualização/Remoção**
  - Atualização/Remoção das informações de um funcionário.
- **Consulta**
  - Consulta das informações de um funcionário.
  - Consulta da evolução do salário de cada funcionário.

Gestão de Clientes

- **Inserção**
  - Novos cliente podem ser adicionados especificando todas as suas informações
- **Atualização/Remoção**
  - Atualização/Remoção de um cliente.

- **Consulta**
  - Consulta das informações de um cliente.
  - Consulta do histórico de compras num dado intervalo de tempo.
  - Consulta do valor total gasto na bomba de cada cliente.

2.2.3 Requisitos de Controlo

Gestão de Stock

- A consulta informações sobre os produtos/combustiveis/fornecedor/encomendas são acessíveis ao admnistrador e aos funcionários.
- A atualização/remoção de produtos/combustiveis só é permitida ao administrador.
- A criação/atualização/remoção de fornecedores só é permitida ao admnistrador.
- A criação/atualização de encomendas só é permitida ao admnistrador.

Gestão de Vendas

- O administrador e funcionário podem emitir novas faturas.
- O administrador e funcionário conseguem consultar faturas já emitidas.
- Apenas o administrador consegue consultar estatísticas sobre as vendas.

Gestão de Funcionários

- Apenas o administrador pode registrar um novo funcionário.
- Apenas o administrador pode atualizar/remover um novo funcionário.
- Apenas o administrador pode consultar informações/estatísticas sobre um funcionário.

Gestão de Clientes

- O administrador e funcionário têm a capacidade registrar um novo cliente.
- O administrador e funcionário podem consultar informações sobre os clientes.
- Apenas o administrador pode consultar estatísticas sobre os clientes.

2.3 Análise e validação geral dos requisitos

A terceira e última fase da metodologia de levantamento de requisitos foca-se na validação dos mesmos. Nesta fase a equipa de trabalho utilizou os seus recursos humanos, nomeadamente o arquiteto e o engenheiro de Bases de Dados, para rever e validar todos os requisitos.

Para confirmar e validar o trabalho desenvolvido, o gestor de projeto reuniu com o gerente e a equipa interna e garantiu que os requisitos cumprem as condições necessárias para descrever o funcionamento da bomba. Durante as várias reuniões que ocorreram nesta fase alguns requisitos foram modificados, outros adicionados e alguns removidos. Alguns resultados destas reuniões são os exemplos seguintes:

- Modificado : O funcionário e administrador podem criar/atualizar encomendas. $\rightarrow$ Apenas o administrador cria/atualiza encomendas.
- Adicionado : Consulta ordenada dos maiores fornecedores.
- Removido : Registro do horário de cada funcionário e quais turnos têm disponibilidade caso seja necessário.

Após a validação por parte do Sr.Domingos, a equipa está confiante que pode avançar para a fase seguinte do projeto - a Modelação Conceptual.

3 Modelação Conceptual

3.1 Metodologia utilizada para a Modelação

A modelação conceptual é uma representação abstrata e de alto nível da estrutura e conteúdo de uma Base de Dados.

A equipa de trabalho a partir da fase anterior "Levantamento e Análise de Requisitos" definiu a seguinte abordagem para a modelação:

- Identificação das Entidades : identificar as entidades do sistema - ou seja, os objetos, conceitos ou eventos que serão armazenados na Base de Dados.
- Identificação dos Relacionamentos: definir os relacionamentos entre entidades. Um relacionamento é caracterizado pela sua Cardinalidade (1:1, 1:N ou N:M) e Participação (Parcil:Total,Total:Total).
- Associação dos Atributos às Entidades e Relacionamentos : definir os atributos, que são as propriedades que descrevem uma entidade ou uma relacionamento-entidade.
- Construção do Modelo Conceptual : utilizando a ferramenta BrModelo constroi-se o diagrama descrito nos procedimentos anteriores.

O resultado da Modelação Conceptual é um diagrama Entidade-Relacionamento , que ilustra graficamente as entidades e suas relações, e assim, resulta numa boa base para modelação lógica da Base de Dados.

3.2 Identificação das Entidades

Analisando os requisitos de descrição do sistema é possível identificar as seguintes entidades em cada uma das áreas de gestão da bomba.

Gestão de Stock

- Produto : representa todos os produtos comercializados na loja, o que inclui os artigos de loja e os combustíveis.

- Fornecedor : representa os fornecedores dos produtos comercializados.
- Encomenda : representa as encomendas feitas aos fornecedores de modo a reestabelecer o stock dos produtos na loja.
- Funcionário : o funcionário que regista as encomendas.

Gestão de Vendas

- Venda : representa as vendas de produtos que são realizadas na bomba, assim como informação adicional.
- Produto : representa os produtos que compõem cada venda.
- Bomba : representa os dados da empresa presente na fatura.
- Funcionário : representa os dados do funcionário que emitiu a fatura.
- Cliente : representa os dados do cliente que realizou a compra.

Gestão de Funcionários

- Funcionário : representa os dados do colaborador.
- Bomba de Combustível : representa os dados do empregador.

Gestão de Clientes

- Cliente : representa os clientes das bombas e os seus dados.
- Venda : representa as vendas associadas ao cliente.

Algumas entidades repetem-se em diferentes áreas da gestão da bomba, isto é uma ocorrência normal, no entanto, de modo a facilitar a modelação é recomendado identificar as entidades únicas, assim sendo, as entidades do sistema são:

- Produto
- Fornecedor
- Encomenda
- Funcionário
- Bomba de Combustível
- Venda
- Cliente

3.3 Identificação dos Relacionamentos

Estabelecidas as entidades do sistema é possível identificar os relacionamentos entre as mesmas.

- Bomba de Combustível - Funcionário [Contrato] (emprega) - 1:N | T:T
  - A bomba e os funcionários estão associados por um contrato laboral e toda a sua informação necessária.
- Funcionário - Venda (regista) - 1:N | P:T
  - Um funcionário regista vendas e a sua informação faz parte da fatura. Um funcionário pode não ter vendas associadas mas uma venda está sempre ligada a um funcionário.
- Venda - Produto [ProdutosVenda] (inclui) - N:M | T:P
  - Uma venda inclui uma lista de produtos e cada produto integra mais do que uma venda. De notar que cada produto na lista tem informação adicional como quantidade, desconto, etc.
- Venda - Bomba de Combustível (associada) - N:1 | T:P
  - Uma fatura inclui informação obrigatória sobre a bomba.
- Venda - Cliente (feita_a) - N:1 | T:P
  - Uma fatura inclui informação obrigatória sobre o cliente, caso não esteja registado no sistema é associado um cliente anónimo.
- Funcionário - Encomenda (emite) - 1:N | P:T
  - O funcionário emite encomendas aos fornecedores, logo esta associado a cada encomenda.
- Fornecedor - Encomenda (entrega) - N:M | P:T
  - Vários fornecedores compõe uma encomenda pois são compostas por diferentes produtos, e um fornecedor entrega várias encomendas ao longo do tempo. De notar que uma encomenda tem de ter obrigatoriamente informação sobre o fornecedor.
- Fornecedor - Produto (fornece) - 1:N | P:T
  - Cada fornecedor fornece uma parte dos produtos comercializados. Cada produto tem de conter esta informação obrigatoriamente.
- Encomenda - Produto [ProdutosEncomenda] (inclui) - N:M | T:P

– Cada encomenda inclui um conjunto de produtos e cada produto integra uma ou mais encomendas, desse modo, o conjunto de produtos tem informação própria como o preço de retalho e quantidade.

3.4 Associação dos Atributos às Entidades e Relacionamentos

Com base nos requisitos levantados, para cada entidade e relacionamento definido foram associados os seguintes atributos:

Entidades

- Produto : Id, Nome, Marca, Categoria, Preço PVP, Stock.
- Fornecedor : Id, Nome, Endereço (Rua, Localidade, Código-Postal, Porta), Contacto (E-mail, Telemóvel(1..n)), Descrição, NIF.
- Encomenda : Id, Data da Encomenda, Data Prevista de Entrega, Data de Entrega, Valor Total, Estado(Pendente, Cancelada, Entrega).
- Funcionário : Id, Nome, Data de Nascimento, Contacto (E-mail, Telemóvel(1..n)), NIF, IBAN.
- Bomba de Combustível : Id, Nome, Endereço (Rua, Localidade, Código-Postal, Porta), Contacto (E-mail, Telemóvel(1..n)), Capital Social, NIF.
- Venda : Nr.Fatura, Data, Valor Total, Desconto Total, Método de Pagamento.
- Cliente : Id, Nome, Data de Registo, NIF, Endereço (Rua, Localidade, Código-Postal, Porta)[opcional], Contacto (E-mail, Telemóvel(1..n))[opcional].

Relacionamentos

- Contrato : Data de Início, Duração, Salário, Horário Semanal, Função.
- ProdutosVenda : Quantidade, Preço, Valor Total , IVA , Desconto.
- ProdutoEncomenda : Quantidade, Preço Retalho , Valor Total.

3.5 Diagramas Entidade-Relacionamento produzido

Os diagramas foram desenhados utilizando uma ferramenta de software denomina BrModelo com a notação de Chen, em anexo a este documento encontram-se os ficheiros. De forma a

manter a linha de pensamento seguida, o grupo dividiu os diagramas em 4 vistas, com cada área de gestão, e a vista geral.

3.5.1 Vista da Gestão de Stock

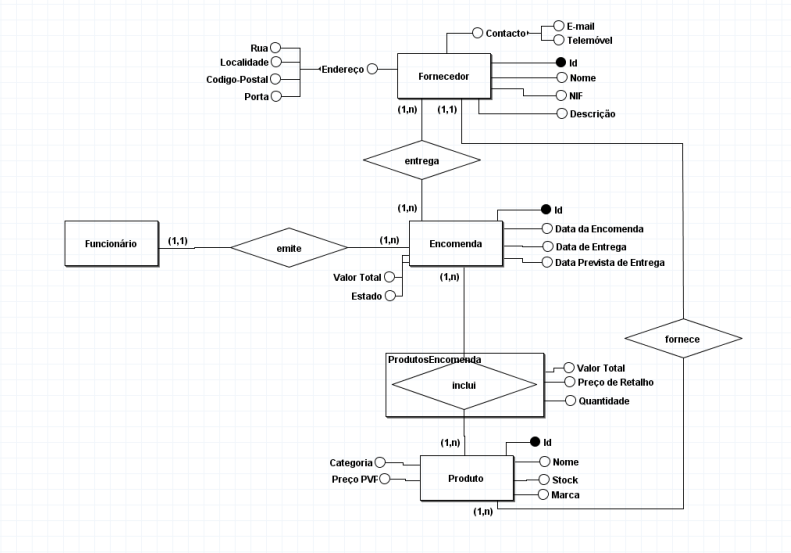

Figura 3.1: Vista da Gestão de Stock

3.5.2 Vista da Gestão de Vendas

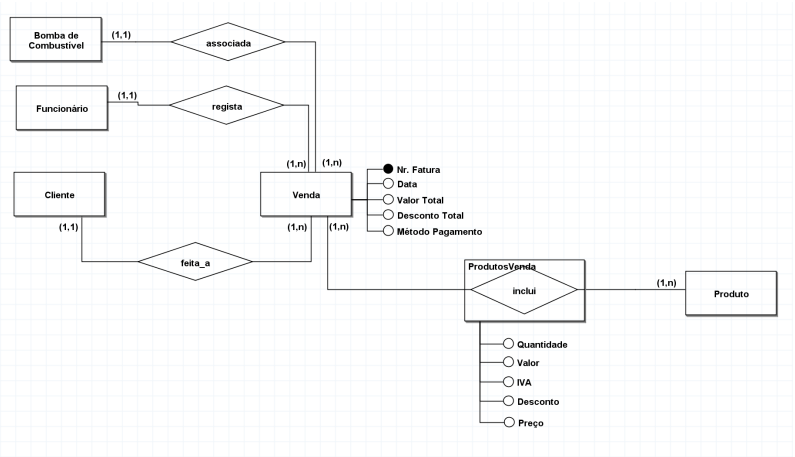

Figura 3.2: Vista da Gestão de Vendas

3.5.3 Vista da Gestão de Funcionários

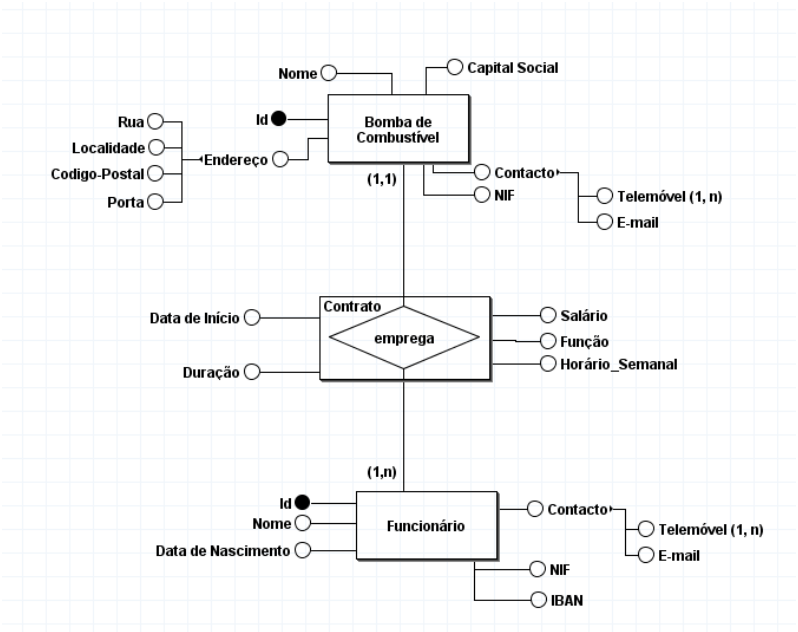

Figura 3.3: Vista da Gestão de Funcionários

3.5.4 Vista da Gestão de Clientes

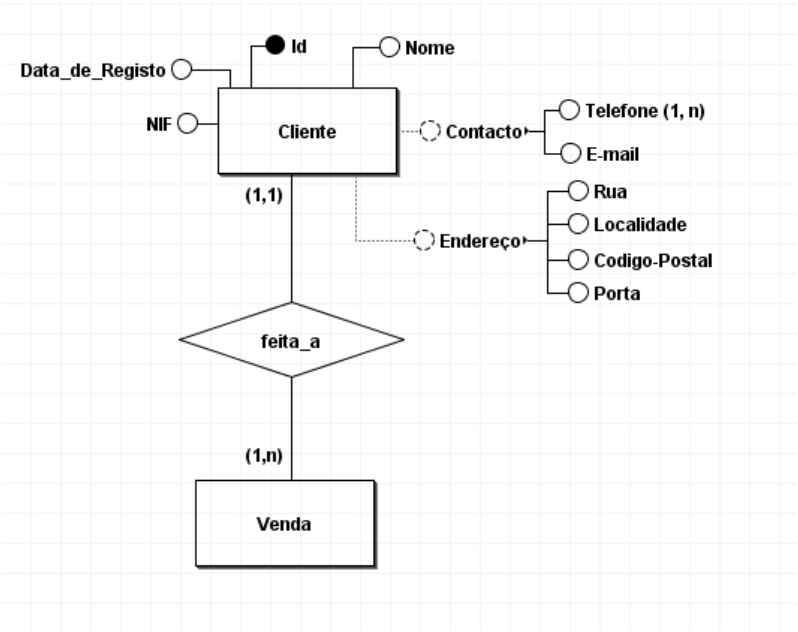

Figura 3.4: Vista da Gestão de Clientes

3.5.5 Vista Geral

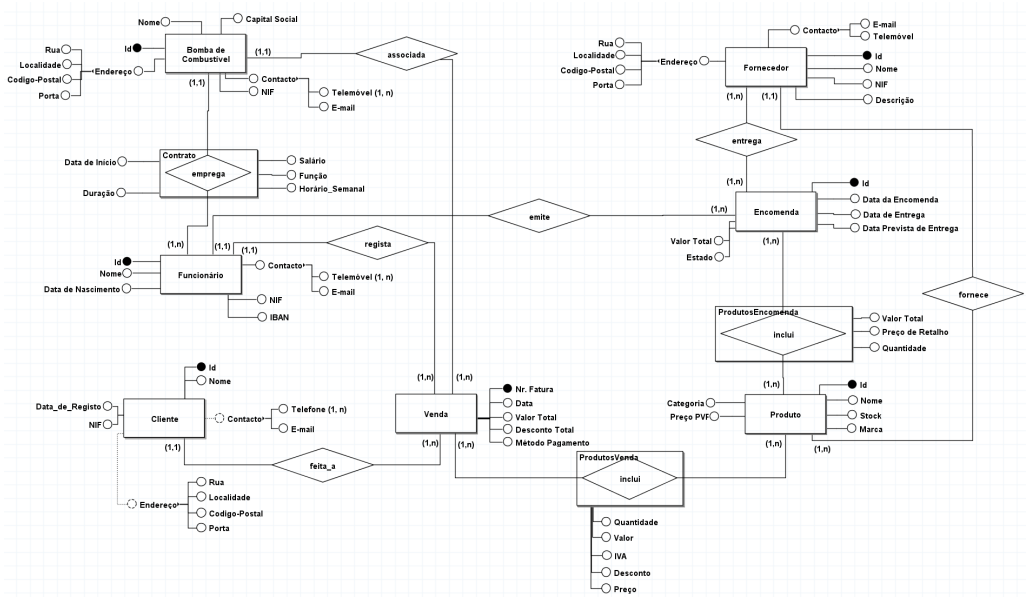

Figura 3.5: Vista Geral

3.6 Análise e Validação do Sistema

Para confirmar e validar o trabalho desenvolvido, o gestor de projeto reuniu com o gerente e a equipa interna e garantiu que o modelo conceptual cumpre os requisitos que descrevem o funcionamento da bomba. Durante as várias reuniões que ocorreram nesta fase, a equipa de trabalho fez algumas modificações com base no feedback recolhido, como por exemplo, a alteração do atributo "Telemóvel"da entidade "Cliente" e "Funcionário"para um atributo multi-valorado, dado que o gerente queria guardar um ou mais números de telemóvel .

4 Modelação Lógica

4.1 Construção modelo de dados lógico

O modelo lógico foi construido com base no Modelo Conceptual. Nesta fase, as entidades, relacionamentos e entidades-relacionamentos serão convertidas em tabelas com atributos e chaves, seguindo a metodologia do modelo relacional. Assim sendo, em cada tabela é necessário identificar a sua chave primária, os seus atributos e as suas chaves estrangeiras (caso existam).

Deste modo, a conversão resultada nas seguintes tabelas:

- **BombaDeCombustivel**
  - Chave Primária: id_bomba : INT
  - Atributos: nome_bomba : VARCHAR(45) | capital_social : INT | nif_bomba : INT | id_contacto : INT | id_endereço : INT
  - Chaves Estrangeiras: contacto_bomba : INT | endereço_bomba : INT
- **Endereço**
  - Chave Primária: id_endereço : INT
  - Atributos: rua : VARCHAR(60) | localidade : VARCHAR(45) | codigo-postal : VARCHAR(10) | porta : VARCHAR(10)
  - Chaves Estrangeiras: ——
- **Contacto**
  - Chave Primária: id_contacto : INT
  - Atributos: e-mail : VARCHAR(45)
  - Chaves Estrangeiras: ——
- **Telefone**

  - Chave Primária: id_contacto : INT
  - Atributos: telefone : VARCHAR(45)
  - Chaves Estrangeiras: id_contacto

- **Cliente**
  - Chave Primária: id_cliente : INT
  - Atributos: data_registo : DATE | nif_cliente : INT | nome_cliente : VARCHAR(45)
  - Chaves Estrangeiras: id_contacto : INT | id_endereço : INT

- **Venda**
  - Chave Primária: id_venda : INT
  - Atributos: data : DATE | valor_total : DECIMAL(9,2) | desconto_total : DECIMAL(6,2) | metodo_pagamento : VARCHAR(45)
  - Chaves Estrangeiras: id_funcionario : INT | id_cliente : INT

- **Encomenda**
  - Chave Primária: id_encomenda : INT
  - Atributos: data_encomenda : DATE | data_prevista_entrega : DATE | data_entrega : DATE | valor_total : DECIMAL(9,2)
  - Chaves Estrangeiras: id_estado : INT | id_funcionario : INT | id_fornecedor : INT

- **Estado**
  - Chave Primária: id_estado : INT
  - Atributos: estado : VARCHAR(45)
  - Chaves Estrangeiras: —

- **Funcionário**
  - Chave Primária: id_funcionario : INT
  - Atributos: nome_funcionario : VARCHAR(45) | data_de_nascimento : DATE | nif_funcionario : INT | iban : VARCHAR(45)
  - Chaves Estrangeiras: id_contacto : INT

- **Produto**
  - Chave Primária: id_produto : INT
  - Atributos: nome_produto : VARCHAR(45) | marca : VARCHAR(45) | stock : DECIMAL(9,3) | preço : DECIMAL(9,2) | desconto : DECIMAL(6,2)
  - Chaves Estrangeiras: categoria : INT | id_fornecedor : INT
- **StockProduto**
  - Chave Primária: id_produto : INT | data : DATE
  - Atributos: stock : DECIMAL(9,3)
  - Chaves Estrangeiras: id_produto : INT
- **Categoria**
  - Chave Primária: id_categoria : INT
  - Atributos: descriçao : VARCHAR(45) | iva : INT
  - Chaves Estrangeiras: —
- **Fornecedor**
  - Chave Primária: id_fornecedor : INT
  - Atributos: nome_fornecedor : VARCHAR(45) | nif_fornecedor : INT | descrição : VARCHAR(45)
  - Chaves Estrangeiras: id_contacto : INT | id_endereço : INT
- **Contrato**
  - Chave Primária: id_contrato : INT | id_bomba : INT | id_funcionario : INT
  - Atributos: data_de_inicio : DATE | salário : DECIMAL (6,2) | cargo : VARCHAR(100) | horario_semanal : INT | duraçao : INT
  - Chaves Estrangeiras: id_bomba : INT | id_funcionario : INT
- **ProdutosVenda**
  - Chave Primária: id_venda : INT | id_produto : INT
  - Atributos: quantidade : DECIMAL(9,2) | valor : DECIMAL(9,2) | preçofinal : DECIMAL(9,2)

  - Chaves Estrangeiras: id_venda : INT | id_produto : INT

- **ProdutosEncomenda**
  - Chave Primária: id_produto : INT | id_encomenda : INT
  - Atributos: quantidade : DECIMAL(9,2) | preço_retalho : DECIMAL(9,2)
  - Chaves Estrangeiras: id_produto : INT | id_encomenda : INT

Como é possível perceber existem bastantes alterações quanto ao modelo conceptual, isto porque um modelo lógico é construido com uma lógica diferente de um modelo conceptual, o que leva a criar tabelas intermédias como Contrato,ProdutosEncomenda,ProdutosVenda, a tabelas de valores multivalorados como o Contacto com os Telefones ou até mudar alguns atributos de posição como é o exemplo do IVA que passou do ProdutosVenda para o Produto. Também se adicionou uma nova tabela StockProduto para manter um *tracking* do stock dos produtos ao longo do tempo.

Estas alterações devem-se ao novo modo de abordar o problema, mas também porque a equipa percebeu que alterando algumas coisas definidas na etapa anterior resultaria num melhor modelo lógico.

4.2 Normalização de Dados

A normalização tem como objetivo principal evitar a redundância de dados, o que resulta num aumento no desempenho do modelo pois evita anomalias durante a inserção, remoção e modificação de registos, garantindo-se assim a integridade dos dados.

Para avaliar a normalização do modelo, é necessário verificar se ele atende a três regras conhecidas como formulas normais.

A Primeira Forma Normal (1FN) estabelece que os atributos devem ser atómicos, ou seja, as tabelas não devem ter valores repetidos e não podem conter atributos multivalorados. Após uma análise simples das tabelas em nosso modelo, constatamos que essa condição é atendida, visto que nos casos do atributos multivalorados, os números de telefone, os valores são separados para uma outra tabela e integrados na tabela Contacto através de uma chave estrangeira. Com uma simples análise às tabelas do nosso modelo, verificamos que tal não acontece, tornando válida a 1FN.

A Segunda Forma Normal (2FN) só pode ser alcançada se a primeira forma normal for satisfeita. Ela determina que os atributos não-chave devem depender exclusivamente da chave primária da tabela. Por exemplo, consideremos o preço de um produto, este depende unicamente do id do produto atribuído na bomba. O mesmo acontece com o restante modelo, validando a 2FN.

Por fim, a Terceira Forma Normal (3FN) exige que, após atender às duas formas normais anteriores, seja verificado que não existem atributos dependentes uns dos outros, ou seja, se há algum atributo que possa ser derivado de outro atributo. No caso do nosso modelo, nenhum atributo é proporcional ou possível de obter a partir de outro, um exemplo para isto não acontecer é o caso da Categoria da qual se deduz o IVA, dessa forma separa-se o IVA e a Categoria numa tabela, desta forma conseguimos que o modelo satisfaça a 3FN.

Após esta análise, podemos afirmar que o modelo está em conformidade com as regras de normalização.

4.3 Modelo Lógico produzido

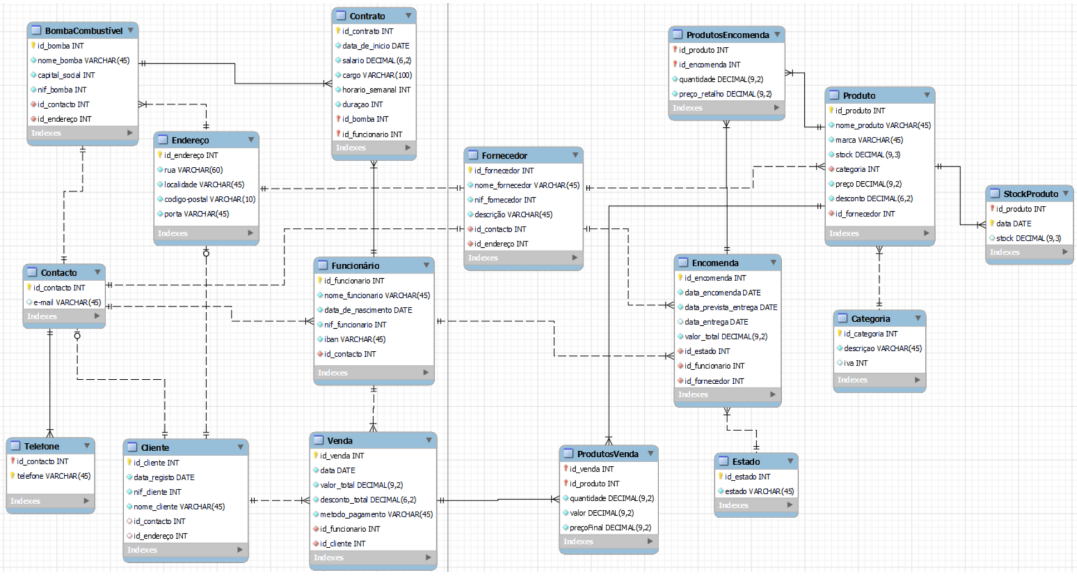

Figura 4.1: Modelo Lógico

4.4 Validação do modelo com interrogações do utilizador

a. Receitas por mês | Seleciona todas as vendas num mês, soma os valores totais e apresenta o resultado.

b. Gastos por mês | Seleciona todas as encomendas num mês, soma os valores totais e apresenta o resultado.

c. Lucros por mês | Deduz os valores dos gastos às receitas daquele mês.

d. Gastos de um Cliente | Inserindo o ID de um cliente, apresenta o valor total gasto pelo cliente.

e. Histórico de faturas de um Cliente | Inserindo o ID de um cliente, deve apresentar todas as vendas associadas a si.

f. Contactos de um Cliente | Inserindo o ID de um cliente, apresenta todos os contactos de um cliente.

g. Top 5 Fornecedores | Apresenta o top 5 de fornecedores segundo o valor das encomendas.

h. Encomendas por Fornecedor | Inserindo o ID de um fornecedor, apresenta todas as encomendas feitas a um fornecedor.

i. Encomendas Pendentes | Seleciona todas as encomendas com o estado "Pendente"e apresenta as suas informações.

j. Stock dos Combustíveis | Apresenta o Stock atual de todos os combustíveis.

k. Stock por Categoria | Inserindo uma categoria, apresenta o Stock atual de todos os produtos de uma categoria.

l. Histórico do Stock de um Produto | Inserindo o id de um produto, apresenta o histórico de stock desse produto.

4.5 Análise e Validação do Sistema

Para confirmar e validar o trabalho desenvolvido, o gestor de projeto reuniu com o gerente e a equipa interna e garantiu que o modelo lógico cumpre os requisitos que descrevem o funcionamento e possível futura expansão da bomba. A aplicação das 3 formas de normalização dos dados garantem a integridade dos dados, as interrogações do utilizador garantem o funcionamento e correta construção da BD para cumprir os requisitos de exploração definidos e a estrutura da BD garante uma possível expansão da BD podem ser possível adicionar novas Bombas, Funcionários, Clientes, Vendas, Encomendas, Fornecedores e Produtos.

5 Implementação Física

5.1 Tradução do esquema lógico para o SGBD escolhido

O grupo escolheu o MySQL como SGBD dado que foi o SGBD escolhido e utilizado durante o decorrer da UC, desta forma, o MySQL Workbench foi a plataforma escolhida pois integra-se bastante bem com o modo de desenvolvimento da BD que o grupo adotou.

Depois de construir e especificar o Modelo Lógico, o grupo utilizou a ferramenta "Database" > "Forward Engineering" do MySQL Workbench para gerar automaticamente o código que cria todas as tabelas. Anexado a este documento encontra-se o ficheiro `create_script.sql` com todo o código SQL obtido.

5.2 Tradução das interrogações do utilizador para SQL

Definidas as interrogações do utilizador objetivo o grupo traduziu-as para sql da seguinte forma

a. Receitas por mês

```sql
CREATE PROCEDURE receitasPorMes (IN mes INT, IN ano INT)
BEGIN
    SELECT MONTH(v.data) AS Mes, YEAR(data) AS Ano, SUM(pv.valor) AS Receita
    FROM BombasBD.Venda v
    INNER JOIN BombasBD.ProdutosVenda pv ON v.id_venda = pv.id_venda
    WHERE YEAR(data) = ano AND MONTH(data) = mes
    GROUP BY MONTH(data), YEAR(data);
END //
DELIMITER ;
```

Figura 5.1: Código SQL correspondente à primeira interrogação

b. Gastos por mês | Seleciona todas as encomendas num mês, soma os valores totais e apresenta o resultado.

```sql
DELIMITER //
CREATE PROCEDURE gastosPorMes (IN mes INT, IN ano INT)
BEGIN
    SELECT MONTH(data) AS Mes, YEAR(data) AS Ano, SUM(valor) AS Gastos
    FROM BombasBD.Gastos
    WHERE YEAR(data) = ano AND MONTH(data) = mes
    GROUP BY MONTH(data), YEAR(data);
END //
DELIMITER ;
```

Figura 5.2: Código SQL correspondente à segunda interrogação

c. Lucros por mês | Deduz os valores dos gastos às receitas daquele mês.

```sql
CREATE PROCEDURE lucrosPorMes (IN mes INT, IN ano INT)
BEGIN
    SELECT MONTH(data) AS Mes, SUM(valor_venda - valor_compra) AS Lucros
    FROM BombasBD.Vendas
    WHERE YEAR(data) = ano AND MONTH(data) = mes
    GROUP BY MONTH(data), YEAR(data);
END //
DELIMITER ;
```

Figura 5.3: Código SQL correspondente à terceira interrogação

d. Gastos de um Cliente | Inserindo o ID de um cliente, apresenta o valor total gasto pelo cliente.

```sql
DELIMITER //

CREATE PROCEDURE GastoPorCliente(IN customerId INT)
BEGIN
    SELECT c.nome_cliente AS NomeCliente, SUM(v.valor_total) AS ValorTotalGasto
    FROM Cliente c
    LEFT JOIN Venda v ON c.id_cliente = v.id_cliente
    WHERE c.id_cliente = customerId
    GROUP BY c.id_cliente;
END //

DELIMITER ;
```

Figura 5.4: Código SQL correspondente à quarta interrogação

e. Histórico de faturas de um Cliente | Inserindo o ID de um cliente, deve apresentar todas as vendas associadas a si.

```
DELIMITER //

CREATE PROCEDURE historico_cliente(IN cliente_id INT)
BEGIN
    SELECT Cliente.nome_cliente AS Nome ,
           Cliente.nif_cliente AS NIF ,
           Venda.data AS Data,
           Venda.valor_total AS Preço,
           Venda.desconto_total AS Desconto
    FROM Venda
    INNER JOIN Cliente ON Venda.id_cliente = Cliente.id_cliente
    WHERE Cliente.id_cliente = cliente_id;
END //

DELIMITER ;
```

Figura 5.5: Código SQL correspondente à quinta interrogação

f. Contactos de um Cliente | Inserindo o ID de um cliente, apresenta todos os contactos de um cliente.

```
DELIMITER //

CREATE PROCEDURE contactos_cliente(IN cliente_id INT)
BEGIN
    SELECT Cliente.nome_cliente AS Nome ,
           Contacto.`e-mail`AS 'E-mail' ,
           GROUP_CONCAT(Telefone.telefone SEPARATOR ', ') AS Telefones
    FROM Cliente
    LEFT JOIN Contacto ON Cliente.id_contacto = Contacto.id_contacto
    LEFT JOIN Telefone ON Contacto.id_contacto = Telefone.id_contacto
    WHERE Cliente.id_cliente = cliente_id
    GROUP BY Cliente.id_cliente;
END //

DELIMITER ;
```

Figura 5.6: Código SQL correspondente à sexta interrogação

g. Top 5 Fornecedores | Apresenta o top 5 de fornecedores segundo o valor das encomendas.

```
DELIMITER //

CREATE PROCEDURE top_5_fornecedores()
BEGIN
    SELECT fornecedor.nome_fornecedor AS 'Fornecedor',
           COUNT(encomenda.id_encomenda) AS 'Nº de Encomendas',
           SUM(encomenda.valor_total) AS 'Total Encomendas'
    FROM fornecedor
    INNER JOIN encomenda ON fornecedor.id_fornecedor = encomenda.id_fornecedor
    GROUP BY fornecedor.id_fornecedor
    ORDER BY SUM(encomenda.valor_total) DESC
    LIMIT 5;
END //

DELIMITER ;
```

Figura 5.7: Código SQL correspondente à sétima interrogação

h. Encomendas por Fornecedor | Inserindo o ID de um fornecedor, apresenta todas as encomendas feitas a um fornecedor.

```
DELIMITER //

CREATE PROCEDURE encomenda_fornecedor(IN fornecedor_id INT)
BEGIN
    SELECT
        encomenda.id_encomenda AS 'Nr Encomenda',
        encomenda.data_prevista_entrega AS 'Data Prevista',
        encomenda.valor_total AS 'Valor da Encomenda',
        fornecedor.nome_fornecedor AS 'Fornecedor',
        estado.estado AS 'Estado'
    FROM
        encomenda
        INNER JOIN estado ON encomenda.id_estado = estado.id_estado
        INNER JOIN fornecedor ON encomenda.id_fornecedor = fornecedor.id_fornecedor
    WHERE
        fornecedor.id_fornecedor = fornecedor_id;
END //

DELIMITER ;
```

Figura 5.8: Código SQL correspondente à oitava interrogação

i. Encomendas Pendentes | Seleciona todas as encomendas com o estado "Pendente"e apresenta as suas informações.

```sql
DELIMITER //

CREATE PROCEDURE encomendas_pendentes()
BEGIN
    SELECT
        encomenda.id_encomenda AS 'Nr Encomenda',
        encomenda.data_prevista_entrega AS 'Data Prevista',
        encomenda.valor_total AS 'Valor da Encomenda',
        fornecedor.nome_fornecedor AS 'Fornecedor',
        estado.estado AS 'Estado'
    FROM
        encomenda
        INNER JOIN estado ON encomenda.id_estado = estado.id_estado
        INNER JOIN fornecedor ON encomenda.id_fornecedor = fornecedor.id_fornecedor
    WHERE
        estado.estado = 'PENDENTE';
END //

DELIMITER ;
```

Figura 5.9: Código SQL correspondente à nona interrogação

j. Stock de Produtos numa Categoria | Inserindo o id da Categoria, apresenta o stock atual de todos os produtos dentro nessa categoria.

```sql
DELIMITER //

CREATE PROCEDURE StockProdutosdaCategoria(IN categoryId INT)
BEGIN
    SELECT p.nome_produto AS NomeProduto, p.stock AS Stock
    FROM Produto p
    INNER JOIN Categoria c ON p.categoria = c.id_categoria
    WHERE c.id_categoria = categoryId;
END //

DELIMITER ;
```

Figura 5.10: Código SQL correspondente à décima interrogação

k. Stock da Categoria | Inserindo uma categoria, apresenta o Stock total da categoria.

```
DELIMITER //
CREATE PROCEDURE stockDaCategoria()
BEGIN
    SELECT c.id_categoria, c.descriçao, SUM(p.stock) AS total_stock
    FROM BombasBD.Categoria c
    INNER JOIN BombasBD.Produto p ON c.id_categoria = p.categoria
    GROUP BY c.id_categoria, c.descriçao;
END //
DELIMITER ;
```

Figura 5.11: Código SQL correspondente à décima primeira interrogação

l. Histórico do Stock de um Produto | Inserindo o id de um produto, mostra o histórico de stock desse produto.

```
DELIMITER //

CREATE PROCEDURE Historico_StockProduto(IN produtoID INT)
BEGIN
    SELECT p.nome_produto AS 'Nome do Produto',
           sp.data AS 'Data',
           sp.stock AS 'Stock'
    FROM StockProduto sp
    INNER JOIN Produto p ON sp.id_produto = p.id_produto
    WHERE  sp.id_produto = produtoID;
END //

DELIMITER ;
```

Figura 5.12: Código SQL correspondente à décima primeira interrogação

5.3 Definição e caracterização das vistas de utilização

A equipa de trabalho definiu vistas sobre a tabelas de forma a obter a informação num melhor formato e mais conciso.

a. Vista do Cliente | Pretende apresentar a informação dos Contactos e Endereços explicitamente, em vez do seu ID.

```
CREATE VIEW ClienteView AS
    SELECT cliente.nome_cliente AS Nome,
           cliente.data_registo AS 'Data Registo',
           cliente.nif_cliente AS NIF ,
           endereço.rua AS Rua,
           endereço.porta AS Porta,
           endereço.localidade AS Localidade,
           endereço.`codigo-postal` AS 'Código-Postal',
           contacto.`e-mail` AS 'E-mail',
           GROUP_CONCAT(Telefone.telefone SEPARATOR ', ') AS telefones
    FROM Cliente
    LEFT JOIN Contacto ON Cliente.id_contacto = Contacto.id_contacto
    LEFT JOIN Endereço ON Cliente.id_endereço = Endereço.id_endereço
    LEFT JOIN Telefone ON Contacto.id_contacto = Telefone.id_contacto
    GROUP BY Cliente.id_cliente;
```

Figura 5.13: Código SQL da Vista do Cliente

b. Vista do Produto | Pretende apresentar a informação da Categoria onde se insere, o IVA e o nome do fornecedor.

```
CREATE VIEW ProdutoView AS
    SELECT  Produto.`id_produto` AS ID,
            Produto.`nome_produto` AS Nome ,
            Produto.`marca` AS  Marca ,
            Produto.`stock` AS Stock ,
            Categoria.`descriçao` AS Categoria,
            Produto.`preço` AS Preço,
            Categoria.`iva` AS IVA,
            Produto.`desconto` AS Desconto,
            Fornecedor.`nome_fornecedor` AS Fornecedor
    FROM Produto
    LEFT JOIN Categoria ON  Produto.categoria  = Categoria.id_categoria
    LEFT JOIN Fornecedor ON  Produto.id_fornecedor = Fornecedor.id_fornecedor
    GROUP BY Produto.id_produto;
```

Figura 5.14: Código SQL da Vista do Produto

c. Vista do Fornecedor | Pretende apresentar a informação dos Contactos e Endereços explicitamente, em vez do seu ID.

```
CREATE VIEW FornecedorView AS
    SELECT fornecedor.nome_fornecedor AS Nome,
           fornecedor.nif_fornecedor AS NIF ,
           endereço.rua AS Rua,
           endereço.porta AS Porta,
           endereço.localidade AS Localidade,
           endereço.`codigo-postal` AS 'Código-Postal',
           contacto.`e-mail` AS 'E-mail',
           GROUP_CONCAT(Telefone.telefone SEPARATOR ', ') AS telefones
    FROM fornecedor
    LEFT JOIN Contacto ON Fornecedor.id_contacto = Contacto.id_contacto
    LEFT JOIN Endereço ON Fornecedor.id_endereço = Endereço.id_endereço
    LEFT JOIN Telefone ON Contacto.id_contacto = Telefone.id_contacto
    GROUP BY fornecedor.id_fornecedor;
```

Figura 5.15: Código SQL da Vista do Fornecedor

d. Vista do Encomenda | Pretende apresentar a informação do estado da encomenda, do funcionário que emitiu e do fornecedor a quem foi feita.

```
CREATE VIEW EncomendaView AS
    SELECT  Estado.estado AS Estado,
            Encomenda.`id_encomenda` AS ID,
            Encomenda.`data_encomenda` AS `Data de Emissão` ,
            Encomenda.`data_prevista_entrega` AS  `Data de Prevista` ,
            Encomenda.`data_entrega` AS `Data de Entrega` ,
            Encomenda.`valor_total` AS Valor,
            Funcionário.`nome_funcionario` AS Funcionário,
            Fornecedor.`nome_fornecedor` AS Fornecedor
    FROM Encomenda
    LEFT JOIN Estado ON  Encomenda.id_estado  = Estado.id_estado
    LEFT JOIN Funcionário ON   Encomenda.id_funcionario = Funcionário.id_funcionario
    LEFT JOIN Fornecedor ON  Encomenda.id_fornecedor = Fornecedor.id_fornecedor
    GROUP BY encomenda.id_encomenda;
```

Figura 5.16: Código SQL da Vista do Encomenda

e. Vista de Combustíveis | Pretende apresentar a informação sobre o stock atual dos combustíveis.

```
CREATE VIEW CombustiveisView AS
    SELECT  Produto.`id_produto` AS ID,
                Produto.`nome_produto` AS Nome ,
                Produto.`marca` AS  Marca ,
                Produto.`stock` AS Stock ,
                Categoria.`descriçao` AS Categoria,
                Produto.`preço` AS Preço,
                Categoria.`iva` AS IVA,
                Produto.`desconto` AS Desconto,
                Fornecedor.`nome_fornecedor` AS Fornecedor
        FROM Produto
        LEFT JOIN Categoria ON  Produto.categoria  = Categoria.id_categoria
        LEFT JOIN Fornecedor ON  Produto.id_fornecedor = Fornecedor.id_fornecedor
        where Categoria.descriçao = 'Combustivel'
        GROUP BY Produto.id_produto;
```

Figura 5.17: Código SQL da Vista de Combustíveis

f. Vista do StockProduto | Pretende apresentar a informação sobre o Histórico do stock do produto com o nome do produto em vez do ID.

```
CREATE VIEW StockView AS
    SELECT p.nome_produto AS 'Nome do Produto',
           sp.data AS 'Data',
           sp.stock AS 'Stock'
    FROM StockProduto sp
    INNER JOIN Produto p ON sp.id_produto = p.id_produto;
```

Figura 5.18: Código SQL da Vista do StockProduto

Quanta ao acesso, à BD o grupo definiu dois tipos utilizadores no requisitos, o 'Gerente' e o 'Funcionário. Desse modo o grupo criou conta para os 3 trabalhadores da loja atribuindo-lhes o seu papel.

1. Criação das Contas

```
CREATE USER 'celso'@'localhost'IDENTIFIED BY 'passCelso';

CREATE USER 'miguel'@'localhost' IDENTIFIED BY 'passMiguel';

CREATE USER 'junior'@'localhost' IDENTIFIED BY 'passJunior';


CREATE ROLE Gerente, Funcionario;

GRANT Gerente TO celso@localhost;
GRANT Funcionario TO junior@localhost;
GRANT Funcionario TO miguel@localhost;
```

Figura 5.19: Código SQL para criar Utilizadores

2. Permissões

```
GRANT SELECT ON bombasbd.produto TO Funcionario;
GRANT SELECT ON bombasbd.fornecedor TO Funcionario;
GRANT SELECT ON bombasbd.encomenda TO Funcionario;
GRANT SELECT, INSERT ON bombasbd.venda TO Funcionario;
GRANT SELECT, INSERT ON bombasbd.cliente TO Funcionario;
GRANT ALL PRIVILEGES ON bombasbd.* TO Gerente;
```

Figura 5.20: Código SQL para gerir permissões

5.4 Cálculo do espaço da bases de dados

De forma a conceber uma estimativa da totalidade de espaço ocupado pela nossa base de dados, através de já conhecidos os tamanhos dos tipos de dados utilizados (como INT, DECIMAL(..), VARCHAR(..), etc) calculamos a ocupação de cada tabela associando, a cada atributo, o espaço ocupado pelo mesmo.

BombasCombustível

Atributos	Tipo	Tamanho (bytes)
id_bomba	INT	4
nome_bomba	VARCHAR(45)	47
capital_social	INT	4
nif_bomba	INT	4
id_contacto	INT	4
id_endereço	INT	4
TOTAL	**-**	**67**

Figura 5.21: Cálculo dos bytes associados à entidade *BombasCombustível*

Endereço

Atributos	Tipo	Tamanho (bytes)
id_endereço	INT	4
rua	VARCHAR(60)	62
localidade	VARCHAR(45)	47
codigo-postal	VARCHAR(10)	12
porta	VARCHAR(10)	12
TOTAL	**-**	**137**

Figura 5.22: Cálculo dos bytes associados à entidade *Endereço*

Contacto

Atributos	Tipo	Tamanho (bytes)
id_contacto	INT	4
e-mail	VARCHAR(45)	47
TOTAL	**-**	**51**

Figura 5.23: Cálculo dos bytes associados à entidade *Contacto*

Telefone

Atributos	Tipo	Tamanho (bytes)
id_contacto	INT	4
telefone	VARCHAR(45)	47
TOTAL	**-**	**51**

Figura 5.24: Cálculo dos bytes associados à entidade *Telefone*

Cliente

Atributos	Tipo	Tamanho (bytes)
id_cliente	INT	4
data_registo	DATE	3
nif_cliente	INT	4
nome_cliente	VARCHAR(45)	47
id_contacto	INT	4
id_endereço	INT	4
TOTAL	**-**	**66**

Figura 5.25: Cálculo dos bytes associados à entidade *Cliente*

Venda

Atributos	Tipo	Tamanho (bytes)
id_venda	INT	4
data	DATE	3
valor_total	DECIMAL(9,2)	6
desconto_total	DECIMAL(6,2)	4
metodo_pagamento	VARCHAR(45)	47
id_funcionario	INT	4
id_cliente	INT	4
TOTAL	**-**	**72**

Figura 5.26: Cálculo dos bytes associados à entidade *Venda*

Encomenda

Atributos	Tipo	Tamanho (bytes)
id_encomenda	INT	4
data_encomenda	DATE	3
data_prevista_entrega	DATE	3
data_entrega	DATE	3
valor_total	DECIMAL(9,2)	6
id_estado	INT	4
id_funcionario	INT	4
id_fornecedor	INT	4
TOTAL	**-**	**31**

Figura 5.27: Cálculo dos bytes associados à entidade *Encomenda*

Estado

Atributos	Tipo	Tamanho (bytes)
id_estado	INT	4
estado	VARCHAR(45)	47
TOTAL	**-**	**51**

Figura 5.28: Cálculo dos bytes associados à entidade *Estado*

Funcionário

Atributos	Tipo	Tamanho (bytes)
id_funcionario	INT	4
nome_funcionario	VARCHAR(45)	47
data_de_nascimento	DATE	3
nif_funcionario	INT	4
iban	VARCHAR(45)	47
id_contacto	INT	4
TOTAL	**-**	**109**

Figura 5.29: Cálculo dos bytes associados à entidade *Funcionário*

Produto

Atributos	Tipo	Tamanho (bytes)
id_produto	INT	4
nome_produto	VARCHAR(45)	47
marca	VARCHAR(45)	47
stock	DECIMAL(9,3)	6
categoria	INT	4
preço	DECIMAL(9,2)	6
desconto	DECIMAL(6,2)	4
id_fornecedor	INT	4
TOTAL	**-**	**122**

Figura 5.30: Cálculo dos bytes associados à entidade *Produto*

StockProduto

Atributos	Tipo	Tamanho (bytes)
id_produto	INT	4
data	DATE	3
stock	DECIMAL(9,3)	6
TOTAL	**-**	**13**

Figura 5.31: Cálculo dos bytes associados à entidade *StockProduto*

Categoria

Atributos	Tipo	Tamanho (bytes)
id_categoria	INT	4
descriçao	VARCHAR(45)	47
iva	INT	4
TOTAL	**-**	**55**

Figura 5.32: Cálculo dos bytes associados à entidade *Categoria*

Fornecedor

Atributos	Tipo	Tamanho (bytes)
id_fornecedor	INT	4
nome_fornecedor	VARCHAR(45)	47
nif_fornecedor	INT	4
descrição	VARCHAR(45)	47
id_contacto	INT	4
id_endereço	INT	4
TOTAL	**-**	**110**

Figura 5.33: Cálculo dos bytes associados à entidade *Fornecedor*

Contrato

Atributos	Tipo	Tamanho (bytes)
id_contrato	INT	4
data_de_inicio	DATE	3
salario	DECIMAL(6,2)	4
cargo	VARCHAR(100)	102
horario_semanal	INT	4
duraçao	INT	4
id_bomba	INT	4
id_funcionario	INT	4
TOTAL	**-**	**129**

Figura 5.34: Cálculo dos bytes associados à entidade *Contrato*

ProdutosVenda

Atributos	Tipo	Tamanho (bytes)
id_venda	INT	4
id_produto	INT	4
quantidade	DECIMAL(9,2)	6
valor	DECIMAL(9,2)	6
preçoFinal	DECIMAL(9,2)	6
TOTAL	**-**	**26**

Figura 5.35: Cálculo dos bytes associados à entidade *ProdutosVenda*

ProdutosEncomenda

Atributos	Tipo	Tamanho (bytes)
id_produto	INT	4
id_encomenda	INT	4
quantidade	DECIMAL(9,2)	6
preço_retalho	DECIMAL(9,2)	6
TOTAL	**-**	**20**

Figura 5.36: Cálculo dos bytes associados à entidade *ProdutosEncomenda*

Tal como aferido pelos "totais"calculados em cada uma das tabelas, a nossa base de dados, sem povoamento ocuparia um total de 1110 bytes. Numa estatística mais realística este valor seria bastante superior, pelo que deveríamos considerar o sucesso do negócio, isto é, as maiores ou menores quantidades de clientes, encomendas, produtos, etc.

5.5 Indexação do Sistema de Dados

Quando um objeto é criado, índices implícitos são automaticamente criados para as colunas com restrição PRIMARY KEY ou UNIQUE. No entanto, também temos a opção de criar nossos próprios índices, conhecidos como índices explícitos. A principal vantagem de criar índices é acelerar as operações de seleção e consulta, como SELECT e WHERE. Porém, é importante notar que esses índices podem desacelerar as operações de inserção e atualização, como INSERT e UPDATE.

Portanto, é recomendado definir índices apenas para colunas que são frequentemente consultadas, mas raramente atualizadas. Os índices são mais eficazes em tabelas com um grande número de entradas, portanto, não é vantajoso definir índices em tabelas com poucas entradas, como as tabelas Funcionários e BombaCombustivel.

Desse modo o grupo escolheu a tabela Fornecedor que é bastantes vezes consultada mas poucas inserções/atualizações ocorrem. Deste modo, correu-se o comando da figura.

```
CREATE INDEX NomeFornecedor  ON Fornecedor(nome_fornecedor);
```

Figura 5.37: Comando para criar o índice NomeFornecedor na coluna nome da tabela Fornecedor

5.6 Procedimentos, Triggers e Funções Implementadas

O grupo implementou um procedimento, cinco triggers e uma função, de modo a automatizar as ações entre tabelas. Isto torna a tabelas mais dinâmica e responsivas, por exemplo, quando se insere um registo numa tabela o valor noutra atualiza.

Procedimento I : Atualizar uma Encomenda para Entregue

```
DELIMITER //
CREATE PROCEDURE AtualizarEncomendaParaEntregue(IN encomenda_id INT)
BEGIN
    DECLARE estado_atual INT;

    SELECT id_estado INTO estado_atual
    FROM Encomenda
    WHERE id_encomenda = encomenda_id;

    IF estado_atual = 2 THEN
        UPDATE Encomenda
        SET id_estado = 3, data_entrega = CURDATE()
        WHERE id_encomenda = encomenda_id;

        SELECT 'Encomenda ENTREGUE.';
    ELSE
        SELECT 'Encomenda já foi dada como ENTREGUE.';
    END IF;
END//
DELIMITER ;
```

Figura 5.38: Procedimento

Trigger I : Apagar Telefones ao apagar Contacto

```
DELIMITER //

CREATE TRIGGER apagar_telefones_trigger
BEFORE DELETE ON `BombasBD`.`Contacto`
FOR EACH ROW
BEGIN
    DELETE FROM `BombasBD`.`Telefone` WHERE id_contacto = OLD.id_contacto;
END //

DELIMITER ;
```

Figura 5.39: Trigger I

Trigger II : Atualiza Stock de Produtos quando Encomenda é Entregue

```
DELIMITER //

CREATE TRIGGER atualizar_stock_EncomendaEntregue
AFTER UPDATE ON Encomenda
FOR EACH ROW
BEGIN
    IF NEW.id_estado = 3 AND OLD.id_estado = 2 THEN
        UPDATE Produto p
        INNER JOIN ProdutosEncomenda pe ON p.id_produto = pe.id_produto
        SET p.stock = p.stock + pe.quantidade
        WHERE pe.id_encomenda = NEW.id_encomenda;
    END IF;
END//

DELIMITER ;
```

Figura 5.40: Trigger II

Trigger III : Atualiza Valor da Venda e Reduz Stock

```
DELIMITER//

CREATE TRIGGER atualiza_VendaeProduto
AFTER INSERT ON ProdutosVenda
FOR EACH ROW
BEGIN
    UPDATE Venda
    SET valor_total = valor_total + NEW.preçoFinal
    WHERE id_venda = NEW.id_venda;

    UPDATE Produto
    SET stock = stock - NEW.quantidade
    WHERE id_produto = NEW.id_produto;
END//

DELIMITER ;
```

Figura 5.41: Trigger III

Trigger IV : Atualiza o Valor Total da Encomenda

```
DELIMITER //

CREATE TRIGGER atualizar_valor_total_Encomenda
AFTER INSERT ON ProdutosEncomenda
FOR EACH ROW
BEGIN
    DECLARE encomenda_id INT;
    DECLARE novo_valor_total DECIMAL(9,2);
    DECLARE estado_encomenda INT;

    SET encomenda_id = NEW.id_encomenda;

    SELECT id_estado INTO estado_encomenda
    FROM Encomenda
    WHERE id_encomenda = encomenda_id;

    IF estado_encomenda = 2 THEN
        SELECT SUM(quantidade * preço_retalho) INTO novo_valor_total
        FROM ProdutosEncomenda
        WHERE id_encomenda = encomenda_id;

        UPDATE Encomenda
        SET valor_total = novo_valor_total
        WHERE id_encomenda = encomenda_id;
    END IF;
END//

DELIMITER ;
```

Figura 5.42: Trigger IV

Trigger V : Automatiza inserção de um produtoVenda calculando valor e preço final

```
DELIMITER //
CREATE TRIGGER inserir_ProdutosVenda
BEFORE INSERT ON `BombasBD`.`ProdutosVenda`
FOR EACH ROW
BEGIN
    DECLARE valor_produto DECIMAL(9,2);

    SET valor_produto = CalculaValorProduto(NEW.id_produto);

    SET NEW.valor = valor_produto;
    SET NEW.preçoFinal = NEW.valor * NEW.quantidade;
END//
DELIMITER ;
```

Figura 5.43: Trigger V

Function I : calcula o valor do produto com o preço, IVA e desconto

```
DELIMITER //
CREATE FUNCTION CalculaValorProduto(productId INT)
RETURNS DECIMAL(9,2)
DETERMINISTIC
BEGIN
    DECLARE preco DECIMAL(9,2);
    DECLARE desconto DECIMAL(6,2);
    DECLARE iva INT;
    DECLARE precoFinal DECIMAL(9,2);

    SELECT Produto.`preço`, Categoria.`iva`, Produto.`desconto`
    INTO preco, iva, desconto
    FROM Produto
    LEFT JOIN Categoria ON Produto.categoria = Categoria.id_categoria
    WHERE Produto.id_produto = productId;

    SET precoFinal = preco * (1 + (iva / 100));
    SET precoFinal = precoFinal - (precoFinal * (desconto / 100));
    RETURN precoFinal;
END//
DELIMITER ;
```

Figura 5.44: Function I

5.7 Plano de segurança e recuperação de dados

Um plano de segurança e recuperação de dados é uma estratégia essencial para proteger informações valiosas e garantir a continuidade dos negócios em caso de incidentes ou desastres. Este envolve a implementação de medidas de segurança, como controlo de acesso, encriptação e backups regulares, além de procedimentos para a recuperação de dados em situações adversas, como falhas de hardware, ataques cibernéticos ou desastres naturais.

Como plano de recuparação de dados o grupo executa um pequeno script:

1. Abrir um Windows command line.
2. Especificar a diretoria do mysqldump utility.,
‘cd /”C:/Program Files/MySQL/MySQL Server 8./bin”‘
3. Criar um dump do MySQL database executando o comando :
mysqldump.exe –user=YourUserName –password=YourPassword –host=localhost –port=3306 –result-file=C:/Users/YourUserName/Desktop/backup/backup.sql bombasBD

Example 1: Plano de Backup dos Dados

Com um plano de segurança e recuperação de dados adequado, as organizações podem minimizar perdas, proteger informações confidenciais e garantir a continuidade das operações, fortalecendo assim sua posição.

6 Implementação do Sistema de Recolha de Dados

6.1 Apresentação e método do sistema

No nosso sistema de recolha de dados, utilizamos python para conectar à nossa base de dados e inserir registos numa tabela específica. Este sistema permite a adição de dados através um ficheiro de formato JSON dado como entrada.

Para estabelecermos a ligação com o servidor MySQL, indicamos o host, a porta, o utilizador e a palavra-passe. Posteriormente a conexão ser estabelecida, é criado um "cursor"para executar a consulta dos dados.

A função principal - "adicionarTabela- recebe como parâmetros o nome da tabela e o caminho para o ficheiro JSON que contém os dados a serem inseridos na respetiva tabela. Esta função abre o ficheiro JSON, carrega os dados e percorre todos os registos contidos nele. Para cada registo são geradas colunas e os seus respetivos valores para inserção correta na tabela em questão. Cada coluna é obtida a partir da "chave"de cada registo enquanto que os valores são obtidos através dos respetivos valores das "chaves".

No final da inserção destes registos é realizado um "commit"das alterações na base de dados. O "cursor"e a conexão com a base de dados são terminados corretamente e o todo o processo deste sistema encerra.

6.2 Implementação do sistema de recolha

Tal como referido anteriormente, a implementação deste sistema de dados é feita em python e os seus recursos, utilizando os módulos "json"para manipular os dados e o "mysql.connector"para interagir com o servidor. O sistema é simples e direto. Utiliza a função "adicionarTabela"para abrir o ficheiro JSON, ler e escrever os registos numa dada tabela. É importante garantir que no ficheiro de entrada, os dados estejam formatados corretamente, assim como a estrutura da tabela ser compatível com os dados a serem inseridos.

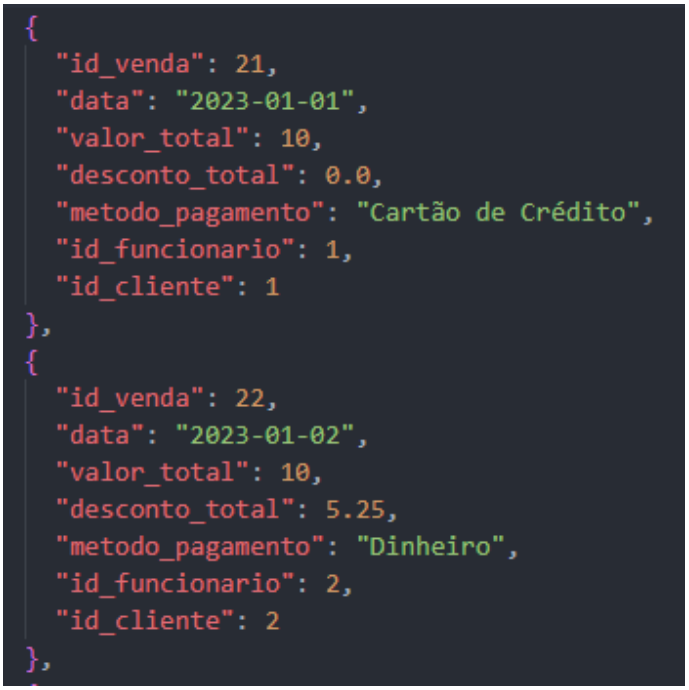

Figura 6.1: Exemplo da formatação utilizada no ficheiro JSON.

6.3 Funcionamento do sistema

Podemos então resumir o funcionamento do nosso sistema de dados:

- È estabelecida a ligação com o servidor MySQL através dos dados necessários (host, porta, utilizador e palavra-passe).
- A função "adicionarTabela"é chamada, com o nome da tabela e o caminho para o ficheiro JSON como argumentos.
- São gerados, para cada registo, colunas e valores.
- Uma consulta é construída através das colunas e respetivos valores e é executada no servidor MySQL através do "cursor".
- Após todas as inserções serem concluídas é realizado um commit das alterações à base de dados.
- Por fim, o "cursor"e a conexão com o servidor são fechados terminando assim o processo.

7 Implementação do Sistema de Painéis de Análise

7.1 Definição e caracterização da vista de dados para análise

O objetivo desta fase é desenvolver um conjunto de painéis que serão utilizados pelo Gerente para ter um entendimento rápido e intuitivo do seu negócio. O grupo utilizou o trabalho desenvolvido da fase 5 com as diferentes vistas e queries como base para os diferentes painéis dos PowerBi. As vistas e queries foram:

- CombustiveisView
- ProdutoView
- EncomendaView
- StockdaCategoria()

7.2 Povoamento das estruturas de dados para análise

Após a implementação física e o povoamento da Base de Dados, o grupo conectou o PowerBi ao MySQL Server de forma a obter os dados para análise. Esta conexão permite que sempre que os valores da BD forem atualizados os dados contidos nos painéis de análise sejam atualizados respetivamente.

7.3 Apresentação e caracterização dos dashboards implementados

A equipa adicionou os seguintes painéis ao dashboard:

- **Lista Stock dos Combustíveis**
- **Lista Stock dos Produtos**
- **Lista de Encomendas**
- **Contador de Clientes**
- **Contador de Vendas**
- **Line Chart Stock Anual de um produto**
- **Pie Chart Stock por Categoria**

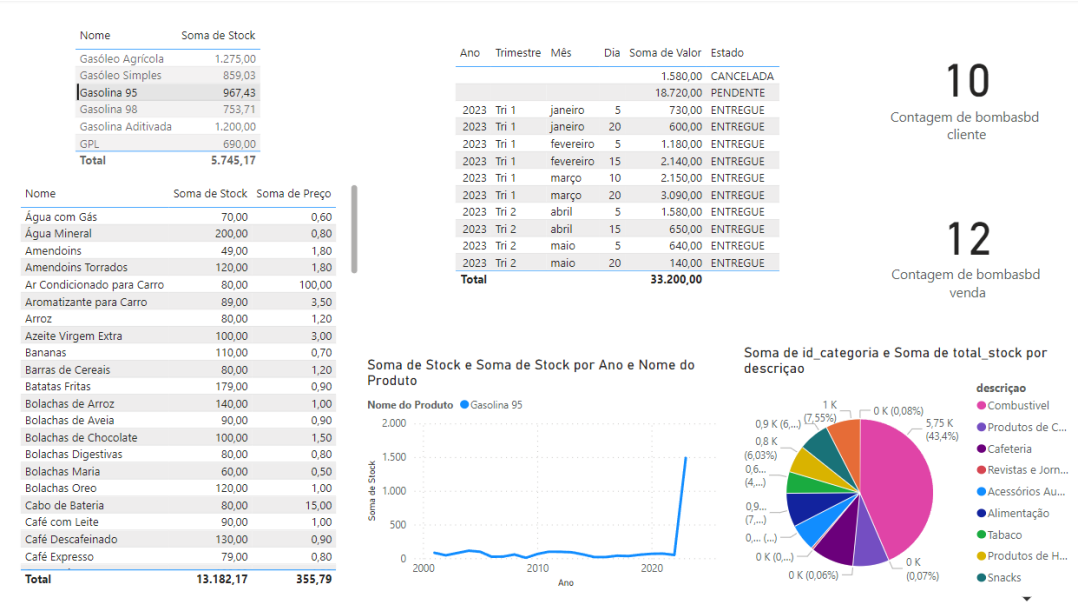

Figura 7.1: DashBoard do PowerBi

8 Conclusões e Trabalho Futuro

Neste relatório, abordámos o projeto de criação de uma base de dados para um posto de abastecimento de combustível, explorando os diferentes aspetos do processo, desde a definição dos requisitos até à implementação física em SQL e à visualização dos dados através do Power BI.

Ao estabelecer requisitos precisos e completos, conseguimos garantir que a base de dados correspondesse às necessidades da bomba e dos seus utilizadores. Isso envolveu a identificação das informações essenciais a serem armazenadas, como os registos de vendas, o stock de combustível e as informações do cliente. Outro ponto crucial foi a criação de um modelo conceptual e lógico sólido. Através do modelo conceptual, mapeámos as entidades e as suas relações, proporcionando uma visão abstrata da estrutura da base de dados. Em seguida, traduzimos esse modelo para um modelo lógico, especificando as tabelas, colunas e chaves estrangeiras necessárias para implementar a base de dados. Esta abordagem permitiu-nos projetar uma estrutura coerente e eficiente, facilitando a manipulação e o acesso aos dados.

Na fase de implementação física em SQL, tratámos da criação de tabelas, índices e restrições, além da escrita de consultas eficientes para inserir, atualizar e eliminar dados. Através do uso adequado de índices, otimização de consultas e gestão adequada do desempenho, conseguimos garantir que o sistema operasse de forma eficaz e responsiva.

Além disso, explorámos a visualização dos dados através do Power BI. Com esta ferramenta, fomos capazes de criar painéis e relatórios interativos que forneceram informações valiosas sobre o desempenho do posto de abastecimento de combustível. Isso permitiu-nos monitorizar as vendas, identificar padrões de consumo, analisar o stock de combustível e tomar decisões informadas sobre o negócio. Apesar de termos obtido resultados satisfatórios com o projeto atual, identificámos áreas que podem ser aprimoradas e exploradas em trabalhos futuros. Uma delas é a possibilidade de expandir a base de dados para incluir outras informações relevantes, como dados de manutenção e gestão de funcionários.

Em suma, este projeto de criação de base de dados para um posto de abastecimento de combustível permitiu-nos aplicar conceitos e técnicas fundamentais no campo da gestão de dados. Através da compreensão dos requisitos, modelação conceptual e lógica, implementação física em SQL e visualização dos dados com o Power BI, conseguimos criar uma solução eficaz para atender às necessidades do negócio. Com as melhorias identificadas e os aprimoramentos futuros, podemos continuar aperfeiçoando esta base de dados e expandindo o seu potencial para uma melhor gestão e tomada de decisões.

Lista de Siglas e Acrónimos

SGBD Sistema de Gestão de Base de Dados

BD Base de Dados

NIF Número de Identificação Fiscal

IBAN International Bank Account Number

SQL Structured Query Language